International Journal of Technological Exploration and Learning (IJTEL)
ISSN: 2319-2135
Volume 2 Issue 5 (October 2013)

A Network Attack Classification for Multi Agent System Using Flow Based Intusion Detection System

K.A.VarunKumar, S.Sibi Chakkaravarthy
Department of Computer Science and Engineering
Vel Tech Dr.RR & Dr.SR Technical University,
Chennai

G.Saravanan, V.Vetriselvan, P.Dharani, C.Aravind
Department of Information Technology,
Department of Electrical & Electronics Engineering
Vel Tech Dr. RR & Dr.SR Technical University, Chennai

Abstract— **Intrusion detection system is precious for protecting the entire network from a large range of threats, modern intrusion detection techniques must cope with not only increase the detection rate but also increase the speed of the network line, signature based ids forced to sample meagerly, increasing the probability of harmful traffic flowing towards the network without inspection subsequently, flow based id is attaining awareness as an effective complement. Basically ids are classified in to network based or host based The independent multi agent design idea is a scalable, striking option for its potential to influence the strengths of both architectures: the broad perspective and visibility into distributed harmful activity provided by network-based ID, and the complete view of the local node provided by host-based ID. This paper develops an architecture for an new multi agent, flow based intrusion detection system. The architecture is designed in two iterations of increasing complexity. These innovative designs are used to find nodes by "reputation "concept that are most effective for classifying harmful network activity. Every system has to design which includes the growth of an innovative classifier that uses multi point evolutionary algorithms to assist in the search for efficient functioning parameter values. Extensive agent simulation framework which highlights the condition under the Reputation System provides a Classification.**

Keywords: Network Attack; Multi Agent; Intrusion Detection; Classification.

I. INTRODUCTION

Discovery, on computer networks, of entities nefarious and profane is the responsibility of an Intrusion Detection System IDS. Techniques for malicious intrusion detection are as diverse as their targets, with associated architectures often characterized in two ways: 1) network-based or host-based; and 2) signature-based or behaviour-based. A signature-based IDS reacts to strings or traffic flowing across its aperture that match a signature in a repository of undesirable bit-patterns. As the signature repository grows over time, it is increasingly very difficult to compare every packet with every signature. The IDS can become overwhelmed with processing and fail to detect many malicious packets. The behaviour-based IDS, on the other hand, reacts to anomalous system activities. Unfortunately, the behaviour-based IDS is designed to notice the symptoms of an already-infected subject. A technique complementary to both these approaches examines statistics related to the inbound and outbound traffic flows This IDS flow-based model is particularly suited for network attacks that make large disturbances in the distributions of these statistics.

Network-based intrusion detection (ID) is typically implemented at the network's gateway. With this system vantage point, it may be able to detect patterns involving multiple hosts that individually would appear innocuous. It is technically challenging or impossible, however, for the network-based IDS to inspect every packet given today's network line speeds, and such an approach inevitably fails to protect individual hosts from every attack.

Host-based ID can be highly responsive to local situations, but has no knowledge of malicious activity that is inherently distributed across the network. A multi agent system paradigm applied to ID attempts to get the best of both of these structural approaches. This paradigm can provide for autonomous, mobile agents that reside transiently at network hosts. Sperotto et al. [16] survey current flow-based ID techniques. In each of 14 systems in their survey, data processing is centralized. In all but four of these systems, data collection is also centralized. In concluding remarks, they point to the research opportunities in "the development of distributed flow-based intrusion detection systems." Hence, this paper discusses a research investigation that develops and analyzes an iteration design of a scalable ID architecture using a multi agent, flow-based intrusion detection system. Nation states loom large in the battle for cyberspace, and international tension is continually on media display. Relatively benign evidence of this is seen in assessments of strategic competitor's network operations capabilities [12]. Of more immediate mediate impact are cases where such capabilities have been put to use. A specific example that recently dominated headlines is the Stuxnet worm. Mafia-style cyber-crime establishments are also rapidly expanding, giving rise to pervasive phishing and botnet herding activities. Even the script kiddies, equipped with powerful and publicly available

II. SCOPE OF INVESTIGATIVE DOMAIN

The generic network of interest to this research is the Autonomous System (AS)-level of the Internet. Each Autonomous System is defined as an administrative domain, comprising within one or more networks of routers, switches, edge devices, and other physical elements, sometimes

involving thousands of addresses. Attacks of interest include those that heavily impact the distribution of the traffic arriving at any given node. Some representatives of this set are: Scans, Distributed Denial of Service Attacks (DDoS) and Worms. Our goal is to develop an effective flow-based, multi agent system for inter-AS network attack classification. It is our hypothesis that we can increase the effectiveness of a flow-based, multi agent network attack classifier by:

- Employing ”reputation” to motivate agents to move when perceived as not providing useful information to peers;
- Decaying the reputation value regardless of ratings received to provide continual impetus for agents to find the best vantage points. The validation of the hypothesis drives specification of the following research objectives and associated evaluation benchmarks:

1) Develop an effective network simulation environment appropriate for the problem scope.
2) Validate the proper functioning of simulated malicious traffic.
3) Validate the proper command, control, and communications in the multi agent intrusion detection system.
4) Study the effects of several factors on classification accuracy

III. INVESTIGATIVE APPROACH

Our research effort consists of two design iterations to ease the complexity of developing ID systems. In each, a network simulation environment, simulation scenarios, and a multi agent system to classify current network activity are developed. Network simulations employ topology and traffic models reflecting associated observations of the Internet. Self organization in the multi agent systems is promoted via the use of a ”reputation” system and evolutionary algorithms for automatic discovery of effective system parameter settings. The first investigative design iteration develops MASNAC (MAS Network Attack Classifier):

- A basic network simulation environment implemented within a Discrete Event Simulation (DES) framework;
- Scenarios for this network environment involving:

1) 10-node networks;
2) Background traffic patterns mimicking the distribution of traffic seen on the actual Internet;
3) DoS attacks;

- A multi agent system for network attack classification employing reputation as a means of governing agent mobility.

The second design iteration, MFIRE (Multi agent, Flow based Intrusion detection system using Reputation and Evolutionary computation), represents an architectural redesign, but incorporates all features from the first iteration and adds many additional features, including:

- A more complex network simulation environment enabling interprocess communication;
- 100-node networks with topological characteristics following those of real-world AS networks;
- Additional malicious scenarios including DDoS attacks, scans and worms;
- A robust communications protocol for the multi agent system, designed to handle transmission losses and other errors.

B. The Use of Reputation

Our reputation system is focused on evaluating agents’ ability to share “useful” information. It is used both to determine the strength of stochastic preference when agents are selected to share information as well as to trigger migration to another node. To elaborate, each agent collects, from its host node, a series of local traffic statistics. Some agents are selected to share summaries of these statistics with other peers. Each agent then makes use of both local as well as shared information to make an individual classification, which is sent to a central entity (controller). Each agent is essentially “voting” on each cycle with respect to the overall classification of the recently observed network activity. The reputation of each agent is affected by whether shared information helped recipients vote in step with the majority, compared with how the recipient would have voted using only local information, resulting in a rating in each case.

C. Internet modeling

The testing environment for our multi agent security research must be both practical and relevant. Simulation can address both issues, but only if the network is modelled properly. The two major issues that arise in attempting to properly model the internet environment are topology and traffic. Approximating the topology of the Internet is discussed in [11]. Empirically derived power laws enable the generation of models of the Autonomous System (AS)-level Internet topology. In [1], Barab´asi and Albert show that power law graphs can be constructed by starting with a random graph and applying incremental growth, enforcing new nodes to prefer connecting to existing, well-connected nodes. With respect to Internet traffic modeling, in recent years a considerable body of research has concluded that self-similar distributions that exhibit fractal behavior (similar statistical properties at any scale) are superior to the models used earlier in pre-Internet telecommunications network traffic modelling (e.g. Poisson) [2], [4], [11], [17]. We use the Pareto model, which is capable of modeling traffic bursts visible at any scale. Such scale-invariant potential for traffic bursts is a commonly observed property of sampled Internet traffic.

D. Pattern Recognition Application

While legitimate traffic is ‘bursty’, flow-based Denial of Service (DoS) traffic saturates, consuming most or all available resources for an extended period of time. Our objective is to detect the disruptive, non-legitimate traffic-generating processes and their targets. To accomplish this, we turn to techniques developed in the general field of pattern recognition [3], [8]. A general description of a pattern recognition system includes the elements depicted in Fig. 1. In [5], [6], each of the pattern recognition system elements are discussed in detail: Pre-processing, Feature generation, and Dimensionality reduction, along with the three fundamental objectives of pattern recognition - Clustering, Classification, and Regression.

IV. MASNAC NETWORK SIMULATION AND MAS DESIGN

Deriving from the principles discussed we present an initial problem domain environment for simulating denial of service attacks on AS-level sections of the Internet, and a multi-agent system for identifying sources and targets of these attacks.

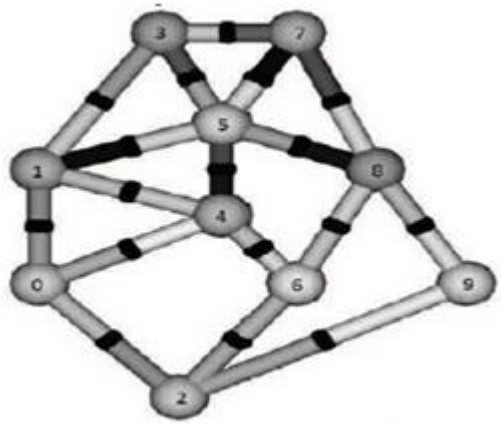

Figure 1. The subject MASNAC network animation

A. A MASNAC Structural Overview

Our MAS system [5] is distributed in the sense that each rule-based agent resides on a different node in the network, making judgments that rely on a combination of 1) direct node observations of traffic statistics and 2) observations requested from other agents. For initial feature selection, our detailed design relies on the multi objective evolutionary algorithm. For agent distribution and provider selection, a reputation system [14] provides performance feedback, motivating agents to find good vantage points in the local neighborhood and helping them assess the value of peer's observations. Our network model is built upon MASON, a discrete-event multi agent simulation engine (see [13]), which was also used for SOMAS (Self-Organized Multi Agent Swarms), our previous agent effort [9]. Formally, we define in [7] a flow-based network with no observable features, and observed denial-of-service attacks A containing disjoint attack class subsets. We find a solution which defines the selected features. Using the selected features, the class mean of each attack class is calculated, and subsequently each attack is classified using a Minimum Euclidean Distance classifier. As we wish to minimize the classification error for all attack classes, the solution is evaluated using multi-objective criteria. We can examine, for example, the MASNAC simulation network depicted in Fig. 2 subjected to attack classes defined by Table I. The numbers in the Source and Target columns of Table I refer to the labels of the nodes in Fig. 2.

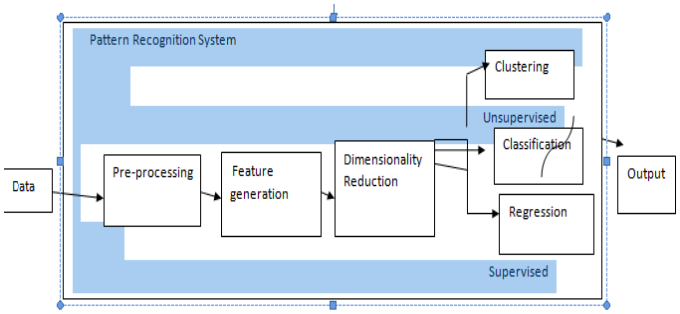

Figure 2. A General Perspective of a Pattern Recognition System

Attack classes are constrained such that each class consists of one DoS process and one target. In particular, all combinations of three DoS source nodes and three target nodes are considered, as well as a "no attack" scenario, for a total of ten attack/activity classes. Note that MASNAC network animation provides temporal performance insight.

TABLE I. MASNAC ATTACK CLASSES

Class	Attack	Target
1	0	7
2	9	7
3	5	7
4	0	3
5	9	3
6	5	3
7	0	2
8	9	2
9	0	2
10	-	-

Figure 3. how agents rate providers of shared observations

TABLE II. CLASSIFICATION RESULT BASED ON OBSERVATION SETS USED

Local Only	Local + Shared	Rating
Same as majority	Same as majority	+0
Same as majority	Different From majority	-0.1
Different From majority	Same as majority	+0.1
Different From majority	Different From majority	-0.05

B. Multi Objective Evolutionary Algorithm (MOEA)

We define features as the mean or standard deviation of the amount of traffic received over a specified period, received from a particular link, and bound for a particular destination. The three periods used are two, five, and ten time steps wide, allowing the system to use a mix of short, medium, and long term statistics. Extended discussion of the MASNAC MOEA approach can be found in [5], [6] where the NSGA-II MOEA was employed. We use an initial base reputation value of 0.5, and Table II to determine how to modify each agent's reputation. As can be seen from Table II, agents, having each selected a single other agent to provide shared observations, rate their providers based on the effect the shared observations had on the classification of the observed activity. For detailed agent operations see [5].

C. MASNAC Classifier Design Evaluatio

The MASNAC MOEA objectives are to minimize classification error of each class and minimize number of features selected. Specific MOEA parameters we used after initial testing are:

- *classes*: 10 network activity scenarios, including nine single-source, single-target DoS attacks and one scenario with only normal (Pareto distributed) traffic

- *Chromosome:* feature selection bit-string. There are 204 bits. These bits can be grouped according to node. For each node, there are 6 * [number of links connected to the node] bits
- population size: 200
- max number of generations: 200,000
- *crossover:* single point, probability 0.9, distribution index

  20.0
- *mutation:* bitflip, probability of flipping each bit in each chromosome is 1 / number of bits = 1/204, distribution index 20.0
- *selection:* binary tournament
- Because we are optimizing across attack classes as indicated in table I, various combinations of the ten attack classes can be visualized as Pareto fronts in 2D or 3D combinations. One of the more interesting is figure 3 which indicates non-dominated values (Pareto front) across three classification errors – note the outliers. Very similar results occur for the other error combinations. The feature sets (Pareto solutions) selected by our MOEA is accomplished in ten dimensions by using a table technique. The associated best feature sets (chromosomes) on this implicit ten dimensional surface are selected. The resulting solutions are feature-selection bit vectors with about 25 features with about 2-3 selected features per node per agentfor classification

V. MFIRE NETWORK SIMULATION AND MAS DESIGN

A new system design is required for autonomous classification of network attacks in a more complex network simulation. This is a natural outcome of progress in ID system design; a more refined network model requires and inspires a more enlightened approach to the design of its defences. Yet, the high-level principles are retained; as before, the second design iteration leverages the multi agent system paradigm with several performance-enhancing details. The agents are designed to be mobile and cooperative in terms of sharing feature observations. Over a series of simulated attacks, the integrated system searches for a 'good' system. The parameters of this reputation distribution of agents via a 'reputation' system are improved a priori via a multi-objective evolutionary algorithm.

A. A. Formal MFIRE Problem Specification

Ultimately we seek to 1) maximize the classification accuracy of all attack classes; and 2) use minimal network bandwidth. This requires an effective spatial distribution of a limited number of agents, where each agent uses a limited number of features to make local classification determinations. These determinations are then shared with a centralized controller. Note that the formal definition provides an unambiguous template for design implementation. The formal MFIRE notation can be found in [5] using the support vector machine (SVM) as the agent-level classification technique.

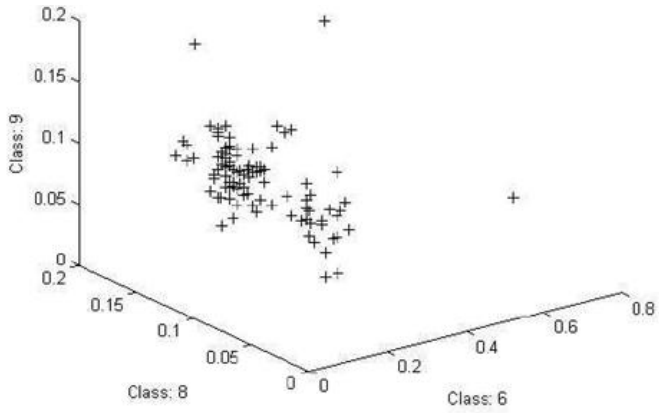

Figure 4. Pareto front classification errors for scenario c6.vs.c8 vs.c9

B. MFIRE Network Simulation and MAS Design Overview:-

The MFIRE package hierarchy involved in the simulation is based upon object-oriented software design. The controlled, one-way dependencies between the visualization layer, the domain layer (labeled 'MFIRE'), and the application layer ('MASON') exhibit a software engineering principle known as *Model-View Separation* This principle states that domain objects should not have direct knowledge of view (UI) objects. It allows the visualization layer to be changed without requiring any changes in the domain or application layer [5]. The domain layer consists of the following groups of classes: Network, Scenarios, Processes, Payloads, Multi agent system, Classification. The MASON discrete event simulation engine package is interfaced which provides many vital facilities for the execution of the simulation as well as the visualization. Network Simulation design objectives for the physical aspect of the AS-level Internet simulation include: 1) Employing a topology representative of the AS level of the Internet; 2) Allowing for multiple scales of AS networks in terms of maximum link distance and 3) Providing visualization/animation facilities to enhance intuitive understanding of simulation execution. We select TopGen for having incorporated some of the most up-to-date observations about the real Internet AS topology. In particular, the Controlled Distance model is used to generate 100-Node networks. Using TopGen, MFIRE sets link lengths by calculating the Euclidean distance and scaling to a specified range. Once the topology is in place, the Floyd Warshall routing algorithm initializes each Node's routing table. The result is that when there is no contention for bandwidth, packets always takes the shortest path to the destination. When the preferred link is saturated with traffic, an alternative link is selected at random. If the alternative link is also saturated, the packet is discarded. The four main traffic patterns are characterized that dominate the simulation. Observe that agents also generate traffic patterns of their own when communicating, but this is normally insignificant in comparison with the patterns described:

1) Simulated Normal traffic follows Pareto distributions [17]

2) Simulated DDoS Traffic Scenarios - Flooding-based Distributed Denial of Service (DDoS) attack

3) Simulated Scan Traffic.

4) Simulated Worm Traffic

C. MFIRE MAS Design

Having completed the design of the simulation framework, the simulated network's topology generation, components, and dominant traffic-generating processes, what remains is the multi agent system charged with distinguishing beign traffic patterns from malicious. The collective activity of the population of agents is tied together at the multi agent system (MAS) level through a controller, which processes the classification decisions ('votes') of individual agents and reports the majority result. The controller also stores and manages agent reputations. Note that the use of a controller makes the multi agent system more vulnerable to disruption, but simplifies the design considerably.

1) MFIRE Design Operational Objectives

Design objectives for our multi agent system include

- Minimize classification error
- Minimize bandwidth
- Provide a robust communications protocol to cope with disruptive traffic patterns The first two competing objectives are achieved by finding a 'good' distribution of a small population of agents. The agents are mobile, and their locational stability is governed by a centralized reputation system managed by the agent controller

  The third objective is elaborated following an overview of the

MAS flow of execution

2) MFIRE Execution Flow Design

The explanation of MFIRE's high-level states is made simpler by assuming agents have been collecting observations from their respective host nodes for nearly a full cycle when it comes time to check in with the Agent Controller. Furthermore, each agent is assumed to have a reputation stored with the Agent Controller. The process is:

- Agents check in with the controller
- Controller makes observation-sharing assignments
- Agents exchange observations per assignments
- Agents calculate and share results with the controller
- Wrap-up:

– The Agent Controller tallies the votes.

– The Agent Controller updates Agent's reputation.

– The Agent Controller sends each Agent a STAY or a MOVE instruction

3) MFIRE Robust Communications

: Each agent synchronizes its clock to match that of the controller and thereby determine the schedule indicating the time of future state transitions.

This allows the Agent to start collecting observations at the same time as others in the system, check in with the Agent Controller at the right time, and so forth. Clock synchronization uses the same basic algorithm employed by the Network Time Protocol (NTP) described in internet RFC 5905. The Agent sends a SYNC message to the Agent Controller. The Agent Controller responds with SYNCREPLY, which contains the time it received the SYNC message as well as the time it sent the SYNCREPLY. The Agent determines the offset by which it needs to adjust its internal clock to match that of the Agent Controller. In the MFIRE Agent, a feature selection filter method based on Bhattacharyya coefficient analysis ranks the features according to a measure of how well they distinguish each class from the others. The top three features from this Bhattacharyya distance-based ranking are preferred. Additionally, we examine the correlation between candidate features. When a pair of features exhibits strong correlation, one of the two is rejected because it is unlikely to contribute useful information beyond what the retained feature provides.

In MFIRE, the agent-level classifier is a Support Vector Machine (SVM). The SVM technique is selected due to its "high generalization performance without the need to add *a priori* knowledge," even in the face of many features. The ability of MFIRE to find effective agent distributions (in terms of location) without external influence is evidence the system possesses attributes of self organization. As summarized by Dempster: "Self-organization refers to exactly what is suggested: systems that appear to organize themselves without external direction, manipulation, or control."

D. Java Implementation

We use the jMetal optimization software package which provides the necessary features for our research: it handles continuous as well as combinatorial problems, is an open framework, and is implemented in Java. The Java implementation facilitates integration with MASON, MASNAC, and MFIRE. The high-level and low-level designs of MFIRE and its network simulation environment are implemented following the respective design closely.

VI. DESIGN OF MASNAC EXPERIMENTS VIA SIMULATION

As reported in [6], [7], the experimental MASNAC design measures and compares the classification accuracy of systems using combinations of the following parameters:

- Number of agents: two, three, and four
- Reputation: not used, used without decay, and used with decay; denoted in some of the tables respectively as **A**, **B, and C**
- Time: four, eight, twelve, and sixteen time steps into the simulation, denoted T1, T2, T3, and T4 Each experiment consists of one combination of the above parameters and 100 attack rounds. For each round, the attack is selected uniformly randomly. Each experiment is run 30 times to achieve a measure of statistical relevance. Note that when reputation is not used, on each round each agent uniformly randomly selects a peer to provide shared observations. Agents in this case do not move from their randomly-selected starting nodes. All links have capacities of 20 units in both directions, which is adequate to handle all but the most severe traffic bursts generated from a Pareto model with $\alpha = 2.0$ and $b = 0$. Nodes have capacities equal to 80% of the maximum amount of data that can arrive in a single time step, which is 20 multiplied by the degree of the node. Any node receiving more than this amount of traffic, on any given time step, shuts

down, muting all resident processes and rendering all connected links unusable; thus, attempting to represent realistic scenarios. The intent of these initial experiments is to provide insight as to classification accuracy with given reputation parameter values and agent communication patterns.

A. . Results and Analysis for MASNAC

Several experimental observations can be made from the statistics reported in [5]. These results are an update of those in [6] based upon more extensive testing. But the qualitative results remain the same. First, regardless of the number of agents used, and regardless of the time step at which the classification is made, using reputation generally outperformed the system that did not. Using reputation with decay generally outperforms using reputation without decay. Decay spurs those agents not receiving negative ratings to nevertheless move to different nodes if over time they do not receive positive ratings, either. It is presumed that this slight increase in exploration yields the measured benefits, but Wilcoxon Rank Sum hypothesis testing reported in [5], [7] (see Table IV) indicates that this cannot be conclusively asserted in all cases. Second, in most cases classification accuracy degraded with increased time. This is because as time passes during a denial of service attack, more nodes have the possibility of shutting down, generating cascading effects as packets are routed differently. With a larger window of time, the variance in the traffic statistics grows, making minimum Euclidean distance based classification more difficult. Third, the standard deviation trends downward from using no reputation, to using reputation without decay, to using reputation with decay, reflecting potentially more consistent results from the latter system. In Table III for the four agents case, bold indicates the 'best' value obtained for the column statistic. In the 'Mean' column only, numbers in italics represent the 'best' values obtained for a particular classification time across the three approaches. In the four agent case, three of the four best mean results are produced by reputation without decay. Reputation with decay achieves the best mean result at T3, and posts the best results for standard deviation in all levels of the time factor. However, Table IV indicates that it is the system using reputation without decay that achieves statistically significant better performance than the system not using reputation for T1 and T2. At T3, using reputation with decay does achieve significantly better performance than using reputation without decay, but neither system significantly outperforms the system that does not use reputation. The primary effect of reputation decay is to induce mobility in stagnate agents, but mobility becomes less important with a higher agent-to-node ratio. A "safe" distance needs to be defined by the operator given the mean statistical numbers. Using improved classification techniques in MFIRE, statistical testing classification accuracies are anticipated to improve.

VII. RESULTS AND ANALYSIS FOR MFIRE

The experimental design for MFIRE focuses on the evaluation benchmarks for the first three previously stated research objectives:

A. Develop an effective network simulation environment appropriate for the problem scope

B. Validate the proper functioning of simulated malicious traffic.

C. Validate the proper command, control, and communications in the multi agent intrusion detection system.

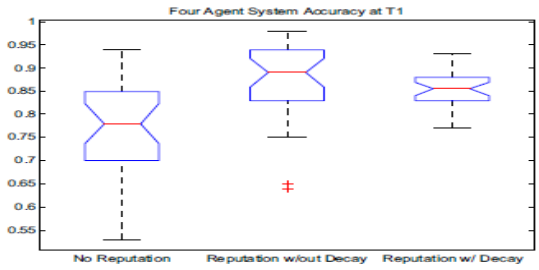

Figure 5. . Classification accuracy of MASNAC under different parameter each bo plot: no reputation; reputation without decay; reputation with decay for Four Agent System

In general, the second iteration design and implementation represented by MFIRE must achieve and cope with greater complexity and realism than the earlier MASNAC-focused iteration. The fourth research objective listed above Study the effects of several factors on classification accuracy, will be addressed in the future, with the intent to demonstrate that the use of a reputation system increases the accuracy of the multi agent network attack classifier within our extended network simulation and multi agent framework MFIRE

D. MFIRE and Simulation Environment Assessment

Assessment of the effectiveness of the MFIRE's network simulation environment includes the following tests, resulting from the research objectives and benchmark specifications.

- Implement a DDoS attack and a ping process
- Implement a UDP-style scan attack
- Implement a worm process and an insecure process

1) MFIRE: DDoS Test

This test creates a DDoS attack scenario involving multiple DoS attackers, a single target, and an ping process that pings the target before and after the attack starts. The output indicates that the ping process sends two sets of pings, with 10 pings in each set. The ping process sends one ping on each timestep and receives 100% response for the first set of pings with no variance in delay. The DDoS then commences from six sources. Figure 5 provides visual indication of what happens in this scenario. In this visualization, each link is divided in half by a black band. For a given node with an attached link, the half of the link directly connected to it shows inbound traffic flow, while the half on the other side of the band shows outbound traffic flow to the adjacent node. The shade of each half link changes according to load: white, on one end of the spectrum, represents zero load; black, on the other end, represents maximum load. Observe that most of the links inbound to the target are at or near maximum capacity. Spillover effects are evident at the adjacent node toward the top of the figure. A short time after the DDoS begins, the ping process starts sending the second set of pings. Responses are received for the first four. The time delay of these responses is significantly longer than what was experienced for the previous

set of pings, with the last taking more than twice as long to return (33 timesteps vs. 16). The cause of this delay is the random rerouting of the packets when the preferred link is full. The rest of the pings fail, for a total loss rate of 60%. This qualitative test demonstrates the desired effect: reliable ping response in the absence of a DDoS attack, and unreliable response in its presence.

2) MFIRE: Scan Test

This experiment demonstrates how a scanning process discovers ports with potentially vulnerable processes. Scan packets are sent to a range of nodes and ports. Insecure processes respond with "OK," while other processes make no response at all to the unexpected packets. For all other ports, the node sends "UNREACHABLE" back to the scanner. In this test the scanning process successfully reports the evidence of insecure and unknown processes distributed among the scanned nodes on random ports.

3) MFIRE: Worm Test

The Worm test creates a worm that makes one propagation attempt (attack) each time step. The environment has four vulnerabilities. Each node has a resident Insecure Process with at least one and up to four of these vulnerabilities. Each vulnerability is assigned a per attack success rate (assuming a matching exploit) of 20%. The worm is armed with exploits for two of the four vulnerabilities, selected randomly. Results indicate increases over time in the number of infected nodes as well as the number of distinct infected nodes making at least one successful attack. In both cases, exponential growth is evident for the majority of the simulation, until growth tapers off because most of the hosts vulnerable to oneof the worm's exploits have already been infected. Our animation for the worm provides a visualization of the infected network [5]. In this scenario, the scale is set to 'regional', allowing link lengths up to 10 units. Black bands separate each of the segments of the link, but one black band divides the link in half. Each half of the link is visually representative of the same concept as used in the DDoS test case: all traffic on each half of the link flows toward the node to which it is directly attached, thus providing a visual separation of the full duplex nature of the link. While no single link in our worm visualization appears saturated as is the case in the DDoS scenario, many links are sustaining high loads at this point in the simulation, which is at approximately t = 120 or near the point at which the growth of the worm slows for lack of uninfected and vulnerable targets. Having demonstrated the desired qualitative effects, this worm attack test is successful. As the third of three complex behaviors, the successful implementation of the worm attack demonstrates the ability of MFIRE's network simulation to support a range of flow-based attacks, satisfying two objectives.

4) . MFIRE Communications and Execution Flow Tests:

The remaining tests involve the communications protocol and execution flow of MFIRE. Eleven test cases, detailed in [5], demonstrate a range of communications failure responses. All test cases are successful.

VIII. CONCLUSION

Our successful research effort develops, in two design iterations, MASNAC and MFIRE, unique multi agent systems designed to engage in flow-based intrusion detection in a distributed way. One of the major innovations of this effort involves the use of reputation as a means of influencing the mobility patterns of the agents. Other innovations include a new network simulation environment for MASON, an associated animated visualization, and the integrated use of multi objective evolutionary algorithms in the multi agent classifier systems.

REFERENCE

[1] M. Becchi, "From Poisson Processes to Self-Similarity: an Survey of Network Traffic Models," 2008. [Online]. Available:http://www.rajjain.com/cse567-06/ftp/traffic models1.pdf

[2] R. O. Duda, P. E. Hart, and D. G. Stork, Pattern Classification (2nd Edition), 2nd ed. Wiley-Interscience, November 2000.

[3] V. Frost and B. Melamed, "Traffic Modeling For Telecommunications Networks," IEEE Communications Magazine, 1994.

[4] D. Hancock, "Multi agent system for flow-based network attack classification," Master's thesis, Air Force Institute of Technology, Wright Patterson AFB, OH, March 2011.

[5] D. L. Hancock and G. B. Lamont, "Multi Agent Systems on Military Networks," in *Proceedings of the IEEE Symposium Series on Computational Intelligence*. IEEE, 2011.

[6] "Reputation in a Multi Agent System for Flow-Based Network Attack Classification," in *Proceedings of the IEEE Symposium on Intelligent Agents*. IEEE, 2011.

[7] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition (Springer Series in Statistics), 2nd ed. Springer, February 2009. [Online]. Available: http://www-stat.stanford.edu tibs/ElemStatLearn/

[8] E. Holloway and G. Lamont, "Self organized multi-agent entangled hierarchies for network security," in *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference*. ACM, 2009, pp. 2589–2596

[9] T.H. Ptacek and T.N. Newsham. Insertion, Evasion and Denial of Service: Eluding Network Intrusion Detection. Technical report, Secure Networks, January 1998.

[10] M. Ranum. Experience Benchmarking Intrusion Detection Systems. NFR Security White Paper, December 2001.

[11] M. Roesch. Snort - Lightweight Intrusion Detection for Networks. In *Proceedings of the USENIX LISA '99 Conference*, November 1999.

[12] S. Northcutt et. al., *Intrusion Signatures and Analysis*. New Riders Press, 2001.

[13] S. Patton, W. Yurcik, and D. Doss, *An Achilles' Heel in Signature-Based IDS: Squealing False Positives in SNORT*. Recent Advances in Intrusion Detection (RAID), Univ. Of\ California-Davis, 2001.

[14] V. Paxson, Bro: A System for Detecting Network Intruders\ in Real-Time, 7th USENIX Security Symposium, 1998.